Theoretical Computer Science 80 (1991) 53-75
Elsevier

An efficient recognizer for the Boolean closure of context-free languages

Stephan Heilbrunner
Universität-GH-Duisburg, FB9 Elektrotechnik, 4100 Duisburg, Germany

Lothar Schmitz
Universität der Bundeswehr, Fakultät für Informatik, 8014 Neubiberg, Germany

Communicated by M. Harrison
Received January 1988
Revised October 1988

Abstract

Heilbrunner, S. and L. Schmitz, An efficient recognizer for the Boolean closure of context-free languages, Theoretical Computer Science 80 (1991) 53-75.

The hci-grammars defined here generate the Boolean closure of the context-free languages. The improved Graham-Harrison-Ruzzo version of Earley's recognition algorithm is adapted to this grammar class and the resulting algorithm is proven correct.

1. Introduction

An *extended regular expression* is a regular expression containing additional operator symbols for complementation (relative to Σ^*) and intersection. Since the regular sets are closed under complementation and intersection, adding these operators does not increase the descriptive power of regular expressions. However, some regular sets are described more concisely and naturally by extended regular sets. For example, the set of all strings not containing two consecutive *b*s is easily described by $[\Sigma^* bb \Sigma^*]$, where the brackets [and] denote complementation. An algorithm for the construction of a deterministic finite automaton from an extended regular expression was given in [4].

Even more concise and natural language descriptions are to be expected if complementation and intersection are used in conjunction with context-free grammars. For this purpose we introduce two new operators, viz. the *complement operator*

and the *intersection operator* by defining

$$L([B]) = \Sigma^* \setminus L(B) \quad \text{and} \quad L(\langle A \cap B \rangle) = L(A) \cap L(B).$$

Since context-free languages are not closed under either complementation or intersection adding these operators effectively increases the descriptive power of context-free grammars. For example, the non-context-free language

$$\{a^n b^n b^n \mid n \in \mathbb{N}\}$$

is described by the following set of rules

$$S \to \langle B \cap D \rangle \qquad L(S) = \{a^n b^n c^n \mid n \in \mathbb{N}\},$$

$$B \to AE \qquad L(B) = \{a^i b^n c^n \mid i, n \in \mathbb{N}\},$$

$$D \to FC \qquad L(D) = \{a^n b^n c^i \mid n, i \in \mathbb{N}\},$$

$$A \to \varepsilon \mid aA \qquad L(A) = \{a^i \mid i \in \mathbb{N}\},$$

$$C \to \varepsilon \mid cC \qquad L(C) = \{c^i \mid i \in \mathbb{N}\},$$

$$E \to \varepsilon \mid bEc \qquad L(E) = \{b^n c^n \mid n \in \mathbb{N}\},$$

$$F \to \varepsilon \mid aFb \qquad L(F) = \{a^n b^n \mid n \in \mathbb{N}\}.$$

All non-palindromes of $\{a, b\}^*$ are given by

$$S \to [A]$$

$$A \to \varepsilon \mid a \mid b \mid aAa \mid bAb.$$

Another, more esoteric example is the set of all non-palindromes containing at least two as which is given by

$$A \to \varepsilon \mid a \mid b \mid aAa \mid bAb \quad \text{all palindromes}$$

$$B \to [A] \quad \text{all non-palindromes}$$

$$C \to \varepsilon \mid aA \mid bA \quad \text{all words}$$

$$D \to CaCaC \quad \text{all words with at least two } a\text{s}$$

$$S \to \langle B \cap D \rangle$$

The non-palindrome examples show that using complementation and intersection *restrictions* of the form "all words satisfying ... but not ..." are very conveniently expressed and combined with other parts of the grammar. One can envision cases where it would be nice to utilize these techniques in practical work as well. For example, when defining an application language, some errors can be characterized as context-free languages themselves; one can incorporate them naturally into the grammar, and then have a non-terminal which stands for a non-error condition. Also, expressing trivial restrictions as in "all strings containing neither two adjacent slashes nor more than 15 slashes" within the grammar might prove useful in practice.

A *recognizer* for a class of grammars is an algorithm which, for any given grammar G and input word w, decides whether w is generated by G or not. In the literature, we find two sub-exponential recognizers for context-free grammars, viz. the Cocke-Younger-Kasami algorithm and Earley's algorithm [1]. A refined version of the Cocke-Younger-Kasami algorithm [6] achieves subcubic asymptotic behaviour but is restricted to grammars in Chomsky normal form. Earley's algorithm requires cubic time but works for unrestricted context-free grammars. Improved versions, termed as "good practical algorithms", are given in [3, 2].

Our goal is to adapt the improved version of Earley's algorithm to context-free grammars which are extended by complement and intersection operators while retaining cubic behaviour. In [2] several potential application areas including natural language processing and extensible programming languages are described. We believe that applications in these areas would benefit from the improved expressiveness and flexibility of our approach. As stated in [5] the Boolean closure of the class of context-free languages is contained properly in the class of context-sensitive languages.

Section 2 contains formal definitions of this grammar class and related notions. In Section 3 we develop the recognition algorithm and illustrate its workings on a simple example. Finally, in Section 4, a detailed correctness proof of the algorithm is given.

2. Hierarchical complement intersection grammars

This section gives the basic definitions. Let N and Σ be sets of non-terminal and terminal symbols as usual. Define the set of *hierarchical complement intersection (hci) symbols* Φ by

$$\Phi = \Sigma \cup N \cup \{[A] \mid A \in N\} \cup \{\langle A \cap B\rangle \mid A, B \in N\}.$$

We establish our conventions for the use of variables in the following table.

Variables in	$a, b, c, \ldots$	$\ldots x, y, z$	$A, B, C, \ldots$	$\ldots X, Y, Z$	$\alpha, \beta, \ldots$
represent elements of	Σ	Σ^*	N	Φ	Φ^*

For hci symbols X we define the *core* of X by

$$\text{core}(a) = \{a\}, \qquad \text{core}(A) = \{A\},$$

$$\text{core}([A]) = \{A\}, \qquad \text{core}(\langle A \cap B\rangle) = \{A, B\}.$$

A quadruple $G = (N, \Sigma, \Pi, S)$ is a *hierarchical complement intersection grammar* (*hci grammar*) if it is a context-free grammar apart from the following two exceptions:

(i) The rules in Π may be of the form $A \to \rho$ where $\rho \in \Phi^*$.

(ii) There are neither self-intersecting nor self-complementing symbols (as defined below).

It will be immediate that the hci property is decidable. The following grammar will be our running example. All paragraphs marked **Example** will refer to this grammar.

Example grammar. The rules are

$$S \to A[B]C \qquad A \to \varepsilon \qquad B \to bD \qquad C \to c\langle E \cap F\rangle c$$
$$D \to A[E]F \qquad E \to \langle F \cap A\rangle \qquad E \to a \qquad F \to Aa$$

We find

$$\Phi \supseteq \{A, B, \ldots, a, c, [B], [E], \langle E \cap F\rangle, \langle F \cap A\rangle, \ldots\}$$

We define a binary relation $\Rightarrow$ on Φ^* by

$$\rho \Rightarrow \sigma \text{ iff } \exists \rho_1, \rho_2, A, \alpha: \rho = \rho_1 A \rho_2 \wedge \sigma = \rho_1 \alpha \rho_2 \wedge A \to \alpha \in \Pi.$$

Example

$$S \Rightarrow A[B]C \Rightarrow A[B]c\langle E \cap F\rangle c \quad \text{and} \quad B \overset{*}{\Rightarrow} b[E]a.$$

Note that hci symbols of the forms $[A]$ and $\langle A \cap B\rangle$ cannot be derived any further. We define the binary relation *uses complement or intersection of* ($\downarrow$) by

$$A \downarrow B \text{ iff } \exists \rho, \sigma, C: A \overset{*}{\Rightarrow} \rho[B]\sigma \vee A \overset{*}{\Rightarrow} \rho\langle B \cap C\rangle\sigma \vee A \overset{*}{\Rightarrow} \rho\langle C \cap B\rangle\sigma.$$

A symbol A is *self-intersecting or self-complementing* if $A \downarrow^+ A$.

Example. The relation $\downarrow$ may be represented by the graph shown in Fig. 1 where the rank column refers to one of the next definitions.

For any G we define the *hierarchy of basic symbols* inductively by

$$N_0 = \Sigma, \qquad N_1 = \{A \mid \neg\exists B: A \downarrow B\} \cup \Sigma,$$
$$N_{i+1} = \{A \mid \exists B \in N_i \setminus N_{i-1}: A \downarrow B \wedge \neg\exists B \notin N_{i-1}: A \downarrow B\} \cup N_i.$$

Note

$$A \downarrow^+ A \text{ iff } A \notin \bigcup_{i \in \mathbb{N}} N_i.$$

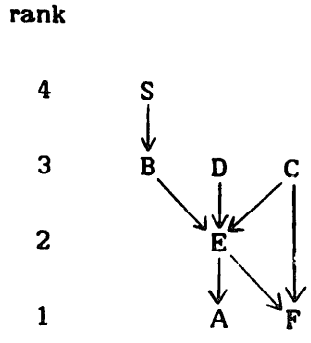

Fig. 1

Example

$$N_0 = \{a, b, c\}, \qquad N_1 = N_0 \cup \{A, F\},$$
$$N_2 = N_1 \cup \{E\}, \qquad N_3 = N_2 \cup \{B, C, D\},$$
$$N_4 = N_3 \cup \{S\}, \qquad N_5 = N_4.$$

We define the *hierarchy constant* h by

$$h = \max\{i \mid N_i \neq N_{i-1} \vee i = 0\}.$$

We define the *ranking function* $\text{rank}: \Phi^* \to \mathbb{N}$ by

$$\text{rank}(a) = 0 \quad \text{for all } a \in \Sigma,$$
$$\text{rank}(A) = i \text{ iff } A \in N_i \setminus N_{i-1}, \quad \text{for all } i,\ i > 0,$$
$$\text{rank}([A]) = \text{rank}(A),$$
$$\text{rank}(\langle A \cap B \rangle) = \max\{\text{rank}(A), \text{rank}(B)\},$$
$$\text{rank}(X_1 \ldots X_n) = \max\{\text{rank}(X_i) \mid 1 \leqslant i \leqslant n\}.$$

We define the *hierarchy of symbols* by $\Phi_i = \{X \mid \text{rank}(X) \leqslant i\}$ and find $\Phi = \Phi_h$ for hci-grammars.

We define the *language* $L(\rho)$ of $\rho \in \Phi^*$ inductively by

$$L(x) = \{x\} \quad \text{for } x \in \Sigma^*,$$

and for strings in $\Phi_{i+1}^* \setminus \Phi_i^*$ by

$$L(A) = \{x \mid \exists \rho \in \Phi_i^* : A \overset{*}{\Rightarrow} \rho \wedge x \in L(\rho)\},$$
$$L([A]) = \Sigma^* \setminus L(A),$$
$$L(\langle A \cap B \rangle) = L(A) \cap L(B),$$
$$L(X_1 \ldots X_n) = L(X_1) \ldots L(X_n).$$

We define an abbreviation by

$$u \in \rho \text{ iff } u \in L(\rho),$$

and show how to decide the question "$\varepsilon \in \rho$?". Our algorithm is a straightforward generalization of the corresponding context-free algorithm. A system of sets $E(r, i)$ is defined for $r, i \in \mathbb{N}$ by

$$E(0, 0) = \{A \mid A \to \varepsilon \in \Pi\},$$
$$E(0, i+1) = \{A \mid \exists \delta \in E(0, i)^* : A \to \delta \in \Pi\} \cup E(0, i),$$
$$E(0, \infty) = \bigcup_{i \in \mathbb{N}} E(0, i),$$

and for $r > 0$

$$E(r, 0) = E(r-1, \infty) \cup \{[A] \mid \text{rank}(A) \leqslant r \wedge A \notin E(r-1, \infty)\}$$
$$\cup \{\langle A \cap B \rangle \mid A, B \in E(r-1, \infty)\},$$
$$E(r, i+1) = \{A \mid \exists \delta \in E(r, i)^* : A \to \delta \in \Pi\} \cup E(r, i),$$
$$E(r, \infty) = \bigcup_{i \in \mathbb{N}} E(r, i).$$

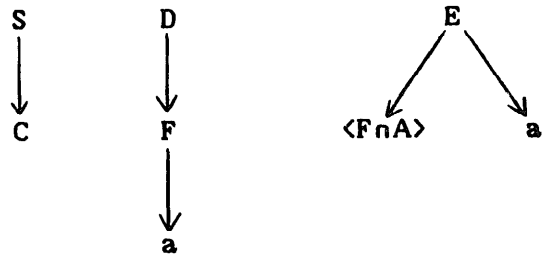

Fig. 2.

It is obvious that for each r the sequence of sets $E(r, i)$, $i = 1, 2, 3, \ldots$ converges. We claim the following.

Lemma 2.1. (i) $\{X \mid \varepsilon \in X\} = E(h, 0)$,
(ii) $\varepsilon \in \rho$ iff $\rho \in E(h, 0)$, *where h is the hierarchy constant.*

Example

$$E(0,0) = E(0,\infty) = \{A\},$$

$$E(1,0) = E(1,\infty) = \{A, [F], \langle A \cap A\rangle\},$$

$$E(2,0) = E(2,\infty) = \{A, [F], \langle A \cap A\rangle, [E]\},$$

$$E(3,0) = E(3,\infty) = \{A, [F], \langle A \cap A\rangle, [E], [B], [C], [D]\},$$

$$E(4,0) = \{A, [F], \langle A \cap A\rangle, [E], [B], [C], [D], [S]\}.$$

Finally, we introduce a relation which ignores ε-symbols by

$$B \Rrightarrow X \text{ iff } \exists \rho, \sigma: B \overset{*}{\Rightarrow} \rho X \sigma \wedge \varepsilon \in \rho \wedge \varepsilon \in \sigma.$$

In order to find all symbols X such that $A \Rrightarrow X$ we define a relation $\mathscr{R}$ by

$$A \mathscr{R} X \text{ iff } \exists \rho, \sigma: A \to \rho X \sigma \wedge \varepsilon \in \rho \wedge \varepsilon \in \sigma$$

and claim the following.

Lemma 2.2. $A \Rrightarrow X$ *iff* $A \mathscr{R}^* X$.

Example. The graph of $\mathscr{R}$ is rather disconnected. It contains the edges as shown in Fig. 2.

3. The recognition algorithm

In this section we shall describe the recognition algorithm. The proof of its correctness will be deferred until the next section.

Definition 3.1. The *left-context grammar* has the rules

$$\{[\![A]\!] \to \alpha[\![B]\!] \mid \exists X, \beta\colon B \in \mathrm{core}(X) \wedge A \to \alpha X \beta \in \Pi\},$$

where bracketed symbols are non-terminals and hci symbols are terminals. We introduce the relation $\Rightarrow$ on bracketed symbols by

$$[\![A]\!] \Rightarrow [\![B]\!] \text{ iff } \exists \rho\colon \varepsilon \in \rho \wedge [\![A]\!] \overset{*}{\Rightarrow} \rho[\![B]\!].$$

Example. The rules of the left context grammar are

$$[\![S]\!] \to [\![A]\!], \quad [\![S]\!] \to A[\![B]\!], \quad [\![S]\!] \to A[B][\![C]\!], \quad [\![B]\!] \to b[\![D]\!],$$
$$[\![C]\!] \to c[\![E]\!], \quad [\![C]\!] \to c[\![F]\!], \quad [\![D]\!] \to [\![A]\!], \quad [\![D]\!] \to A[\![E]\!],$$
$$[\![D]\!] \to A[E][\![F]\!], \quad [\![E]\!] \to [\![F]\!], \quad [\![E]\!] \to [\![A]\!], \quad [\![F]\!] \to [\![A]\!].$$

The graph of the relation $\Rightarrow$ is the transitive closure of the graph shown in Fig. 3.

Definition 3.2. An *item* $A \to \alpha.\beta$ denotes a triple (A, α, β) which satisfies $A \to \alpha\beta \in \Pi$. The *parsing predicate* is defined by

$$P(u, v, A \to \alpha.\beta) \text{ iff } \exists \rho\colon [\![S]\!] \overset{*}{\Rightarrow} \rho[\![A]\!] \wedge u \in \rho \wedge v \in \alpha.$$

The *parsing set* $H(u, v)$ is defined by

$$H(u, v) = \{A \to \alpha.\beta \mid P(u, v, A \to \alpha.\beta)\}.$$

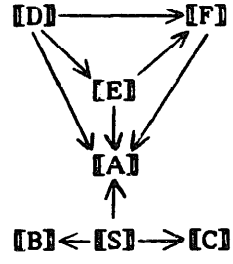

Fig. 3.

We note the difference between $\Rightarrow$ and $\overset{*}{\Rightarrow}$ and compute

$$H(\varepsilon, \varepsilon) = \{A \to \alpha.\beta \mid [\![S]\!] \Rightarrow [\![A]\!] \wedge \varepsilon \in \alpha\}.$$

Example

$$[\![S]\!] \Rightarrow A[\![B]\!] \Rightarrow Ab[\![D]\!] \Rightarrow AbA[E][\![F]\!] \Rightarrow AbA[E][\![A]\!],$$
$$P(b, \varepsilon, D \to A[E].F), \quad P(b, bcbcc, D \to A[E].F),$$
$$P(b, a, D \to A[E]F.), \quad P(b, bcbcca, D \to A[E]F.),$$
$$H(\varepsilon, \varepsilon) = \{S \to .A[B]C, S \to A.[B]C, S \to A[B].C,$$
$$A \to \varepsilon., B \to .bD, C \to .c\langle E \cap F\rangle c\}.$$

Obviously,

$$w \in S \text{ iff } \exists \sigma\colon \quad S \to \sigma . \in H(\varepsilon, w).$$

Hence, we are aiming at an algorithm which computes $H(\varepsilon, w)$. We shall present such an algorithm now and prove its correctness in the next section. Our algorithm is based on the computation of all sets $H(u, v)$ such that $uv = w$. The computation order of the sets $H(u, v)$ is not quite obvious and we will explain it first. For expository purposes we insert the parsing sets for any given word $w = a_1 a_2 \ldots a_n$ into an upper triangular matrix $\mathfrak{H} = (\mathfrak{h}_{ij})$ by assigning the set $H(a_1 \ldots a_i, a_{i+1} \ldots a_j)$ to position (i, j).

The intention of Fig. 3a is to show that $\mathfrak{h}_{ij}$ has *left context* $a_1 \ldots a_i$ and *covers* $a_{i+1} \ldots a_j$. For the computation of $\mathfrak{h}_{ij}$ three cases have to be considered:

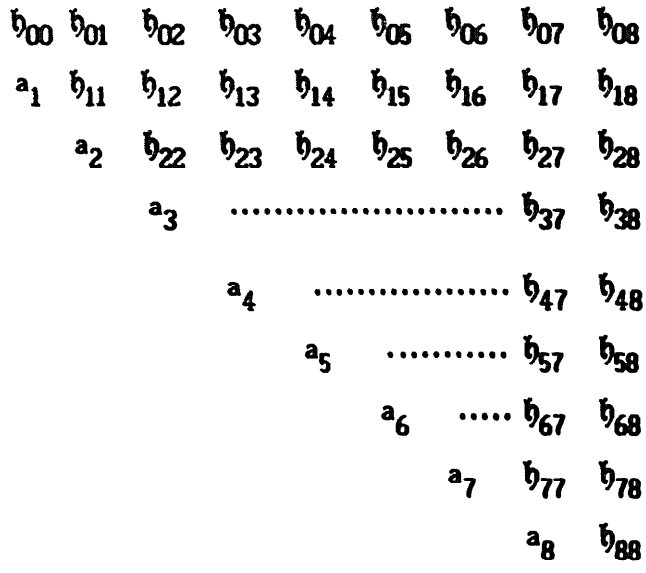

Fig. 3a.

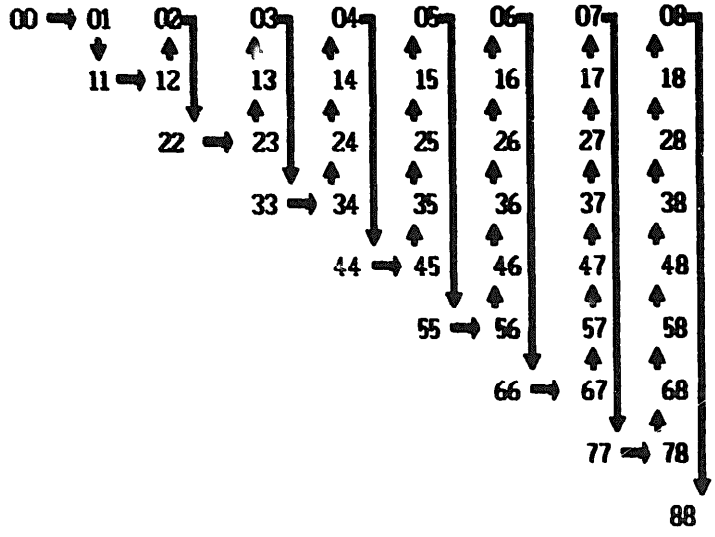

Fig. 3b.

- $\mathfrak{h}_{ij}$ is on the main diagonal, i.e. $j = i$; in this case $\mathfrak{h}_{ij}$ will be seen to depend on all $\mathfrak{h}_{kj}$, $k < i$, i.e. on all elements above $\mathfrak{h}_{ij}$;
- $\mathfrak{h}_{ij}$ is one element to the right of the main diagonal, i.e. $j = i+1$; in this case $\mathfrak{h}_{ij}$ will be seen to depend only on $\mathfrak{h}_{ii}$, i.e. on the diagonal element to its left;
- $\mathfrak{h}_{ij}$ is at least two places off the main diagonal, i.e. $j > i+1$; in this case $\mathfrak{h}_{ij}$ will be seen to depend on all $\mathfrak{h}_{kj}$, $k < j$, and on all $\mathfrak{h}_{ik}$, $k > i$ In other words, all the elements in the same row to the left of $\mathfrak{h}_{ij}$ and all the elements in the same column below $\mathfrak{h}_{ij}$ are needed.

These dependences suggest the computation order which is outlined in Fig. 3b. Figure 3b implies the basic loop structure of our algorithm.

Algorithm 3.3

Initialization: compute $\mathfrak{h}_{00} = H(\varepsilon, \varepsilon)$;
for $j = 1, 2, 3, \ldots, n$ **loop**
  off-diagonal: compute $\mathfrak{h}_{j-1,j} = H(a_1 \ldots a_{j-1}, a_j)$;
  for $i = j-2, j-3, \ldots, 0$ **loop**
    off-off-diagonal: compute $\mathfrak{h}_{ij} = H(a_1 \ldots a_i, a_{i+1} \ldots a_j)$;
  end loop;
  on-diagonal: compute $\mathfrak{h}_{jj} = H(a_1 \ldots a_j, \varepsilon)$;
end loop;

We shall now detail the four compute statements of this algorithm. Concerning the Initialization statement we recall

$$H(\varepsilon, \varepsilon) = \{A \to \alpha.\beta \mid [\![S]\!] \Rightarrow [\![A]\!] \wedge \varepsilon \in \alpha\}$$

so that the computation of $\mathfrak{h}_{00}$ is straightforward.

Example

$$\mathfrak{h}_{00} = H(\varepsilon, \varepsilon) = \{S \to .A[B]C,\ S \to A.[B]C,\ S \to A[B].C,$$
$$A \to \varepsilon.,\qquad B \to .bD,\qquad C \to .c\langle E \cap F\rangle c\}.$$

Before turning to the next compute statement we define an operator on item sets.

Definition 3.4. The *level t reduction operator* $*_t$ on sets L, M of items is defined by

$$\begin{aligned} L *_t M = \{&A \to \alpha B\beta.\sigma \mid A \to \alpha.B\beta\sigma \in L \wedge \operatorname{rank}(B) = t \wedge \varepsilon \in \beta \\ &\wedge \exists D, \delta\colon B \Rightarrow D \wedge D \to \delta. \in M\} \\ \cup \{&A \to \alpha\langle B \cap C\rangle\beta.\sigma \mid A \to \alpha.\langle B \cap C\rangle\beta\sigma \in L \\ &\wedge \operatorname{rank}(\langle B \cap C\rangle) = t \wedge \varepsilon \in \beta \\ &\wedge \exists D, \delta\colon B \Rightarrow D \wedge D \to \delta. \in M \wedge \exists D, \delta\colon C \Rightarrow D \wedge D \to \delta. \in M\} \\ \cup \{&A \to \alpha[B]\beta.\sigma \mid A \to \alpha.[B]\beta\sigma \in L \wedge \operatorname{rank}([B]) = t \wedge \varepsilon \in \beta \\ &\wedge \neg\exists D, \delta\colon B \Rightarrow D \wedge D \to \delta. \in M\}. \end{aligned}$$

The Off-diagonal compute statement may now be refined as

— Off-diagonal: compute $\mathfrak{h}_{j-1,j} = H(a_1 \ldots a_{j-1}, a_j)$;
$M := \{A \to \alpha a_j \beta.\sigma \mid A \to \alpha.a_j\beta\sigma \in \mathfrak{h}_{j-1,j-1} \wedge \varepsilon \in \beta\}$;
for $t = 1, 2, \ldots, h$ **loop**
  $M := \mathfrak{h}_{j-1,j-1} *_t M$;
end loop;
$\mathfrak{h}_{j-1,j} := M$;

Example. Let *cac* be the input word. For $j = 1$ we have $\mathfrak{h}_{01} = H(\varepsilon, c)$. We recall

$$\mathfrak{h}_{00} = H(\varepsilon, \varepsilon) = \{S \to .A[B]C,\ S \to A.[B]C,\ S \to A[B].C,$$
$$A \to \varepsilon.,\qquad B \to .bD,\qquad C \to .c\langle E \cap F\rangle c\}$$

and enter the loop with

$$M = \{C \to c.\langle E \cap F\rangle c\}.$$

Note that M is augmented in the loop although it does not contain "complete" items (i.e. items with a rightmost position of the dot). For $t = 3$ we find $S \to A.[B]C \in \mathfrak{h}_{00}$, rank$(B) = 3$, and

$$M = \{C \to c.\langle E \cap F\rangle c,\ S \to A[B].C\},$$

which is the final value for M. The computation results in

$$\mathfrak{h}_{01} = H(\varepsilon, c) = \{C \to c.\langle E \cap F\rangle c,\ S \to A[B].C\}.$$

We continue our refinement effort with the On-diagonal compute statement because it will be executed next in our running example.

— On-diagonal: compute $\mathfrak{h}_{jj} = H(a_1 \ldots a_j, \varepsilon)$;
$M := \emptyset$;
for $k = 0, 1, \ldots, j-1$ **loop**
  $M := M \cup \{A \to \rho.\sigma \mid \varepsilon \in \rho \wedge \exists B, \alpha, X, \beta, C:$
    $B \to \alpha.X\beta \in \mathfrak{h}_{kj} \wedge C \in \text{core}(X) \wedge [\![C]\!] \Rightarrow [\![A]\!]\}$;
end loop
$\mathfrak{h}_{jj} := M$;

Example. Computation of $\mathfrak{h}_{11}$ is next. Because of $j = 1$ the loop is executed only once and we obtain

$$\mathfrak{h}_{11} = H(c, \varepsilon) = \{A \to \varepsilon.,\ C \to .c\langle E \cap F\rangle c,\ E \to .\langle F \cap A\rangle,\ E \to .a,\ F \to .Aa,\ F \to A.a\}.$$

Now, we may augment j by 1 and compute another off-diagonal element

$$\mathfrak{h}_{12} = H(c, a) = \{E \to a.,\ F \to Aa.\}.$$

We need two more operators before we can refine the remaining compute statement of our algorithm.

Definition 3.5. The (*unrestricted*) *reduction operator* $*$ on sets L, M of items is defined by

$$\begin{aligned} L * M = \{&A \to \alpha B\beta.\sigma \mid A \to \alpha.B\beta\sigma \in L \wedge \varepsilon \in \beta \\ &\wedge \exists D, \delta\colon B \Rightarrow D \wedge D \to \delta. \in M\} \\ \cup \{&A \to \alpha\langle B \cap C\rangle\beta.\sigma \mid A \to \alpha.\langle B \cap C\rangle\beta\sigma \in L \wedge \varepsilon \in \beta \\ &\wedge \exists D, \delta\colon B \Rightarrow D \wedge D \to \delta. \in M \wedge \exists D, \delta\colon C \Rightarrow D \wedge D \to \delta. \in M\} \\ \cup \{&A \to \alpha[B]\beta.\sigma \mid A \to \alpha.[B]\beta\sigma \in L \wedge \varepsilon \in \beta \\ &\wedge \neg\exists D, \delta\colon B \Rightarrow D \wedge D \to \delta. \in M\}. \end{aligned}$$

The *shift operator* $+$ on a set of items and a terminal symbol is defined by

$$L + a = \{A \to \alpha X\beta.\sigma \mid X \Rightarrow a \wedge \varepsilon \in \beta \wedge A \to \alpha.X\beta\sigma \in L\}.$$

The distinction between the operators $*_t$ and $*$ and their respective uses will be motivated in the discussion on "non-trivial and trivial splits", following Lemma 4.3.

The loop containing the off-off-diagonal computations may now be refined as

for $i = j-2, j-3, \ldots, 0$ **loop**
  — off-off-diagonal: compute $\mathfrak{h}_{ij} = H(a_1 \ldots a_i, a_{i+1} \ldots a_j)$;
  $M := \mathfrak{h}_{ij-1} + a_j$;
  for $k = i+1, i+2, \ldots, j-1$ **loop**
    $M := M \cup \mathfrak{h}_{ik} * \mathfrak{h}_{kj}$;
  end loop;
  for $t = 1, 2, \ldots, h$ **loop**
    $M := M \cup \mathfrak{h}_{ii} *_t M$;
  end loop;
  $\mathfrak{h}_{ij} := M$;
end loop;

Example. Now, we are ready to tackle $\mathfrak{h}_{02} = H(\varepsilon, ca)$. Recall $j = 2$, so that the body of the above loop is executed only once for $i = 0$. We compute $M = \emptyset$ before entering the k-loop. In the k-loop, k may assume just one value $k = 1$ and we obtain $M = \{C \to c\langle E \cap F\rangle.c\}$ when leaving the k-loop. Since M does not contain complete items the only item of relevance in $\mathfrak{h}_{00}$ for the t-loop is $S \to A.[B]C$ and $S \to A[B].C$ is added to M. We arrive at entry $\mathfrak{h}_{02} = H(\varepsilon, ca)$ of the matrix $\mathfrak{H}$ as given in Fig. 3c. Note $S \to A[B]C. \in \mathfrak{h}_{03}$ so that $cac \in L(S)$.

We close this section by combining all the refinements into the following algorithm.

Algorithm 3.6

— Initialization: compute $\mathfrak{h}_{00} = H(\varepsilon, \varepsilon)$;

	0	1	2	3
0	$S \to .A[B]C$ $S \to A.[B]C$ $S \to A[B].C$ $A \to \varepsilon.$ $B \to .bD$ $C \to .c\langle E \cap F\rangle c$	$C \to c.\langle E \cap F\rangle c$ $S \to A[B].C$	$S \to A[B].C$ $C \to c\langle E \cap F\rangle.c$	$S \to A[B].C$ $S \to A[B]C.$ $C \to c\langle E \cap F\rangle c.$
1		$A \to \varepsilon.$ $C \to .c\langle E \cap F\rangle c$ $E \to .\langle F \cap A\rangle$ $E \to .a$ $F \to .Aa$ $F \to A.a$	$E \to a.$ $F \to Aa.$	
2			$C \to .c\langle E \cap F\rangle c$	$C \to c.\langle E \cap F\rangle c$
3				$A \to \varepsilon.$ $C \to .c\langle E \cap F\rangle c$ $E \to .\langle F \cap A\rangle$ $E \to .a$ $F \to .Aa$ $F \to A.a$

Fig. 3c. Matrix $\mathfrak{H}$ for *cac*.

$\mathfrak{h}_{00} := \{A \to \alpha.\beta \mid [\![S]\!] \Rightarrow [\![A]\!] \wedge \varepsilon \in \alpha\}$;
for $j = 1, 2, 3, \ldots, n$ **loop**
—off-diagonal: compute $\mathfrak{h}_{i-1,j} = H(a_1 \ldots a_{j-1}, a_j)$;
$M := \{A \to \alpha a_j \beta.\sigma \mid A \to \alpha.a_j\beta\sigma \in \mathfrak{h}_{j-1,j-1} \wedge \varepsilon \in \beta\}$;
for $t = 1, 2, \ldots, h$ **loop**
$M := M \cup \mathfrak{h}_{j-1,j-1} *_t M$;
end loop;
$\mathfrak{h}_{i-1,j} := M$;
for $i = j-2, j-3, \ldots, 0$ **loop**
— off-off-diagonal: compute $\mathfrak{h}_{ij} = H(a_1 \ldots a_i, a_{i+1} \ldots a_j)$;
$M := \mathfrak{h}_{ij-1} + a_j$;
for $k = i+1, i+2, \ldots, j-1$ **loop**
$M := M \cup \mathfrak{h}_{ik} * \mathfrak{h}_{kj}$;
end loop;
for $t = 1, 2, \ldots, h$ **loop**
$M := M \cup \mathfrak{h}_{ii} *_t M$;
end loop;
$\mathfrak{h}_{ij} := M$;
end loop;
— on-diagonal: compute $\mathfrak{h}_{ij} = H(a_1 \ldots a_j, \varepsilon)$;

$M := \emptyset$;
for $k = 0, 1, \ldots, j-1$ **loop**
$M := M \cup \{A \to \rho.\sigma \mid \varepsilon \in \rho \wedge \exists B, \alpha, X, \beta, C:$
$B \to \alpha.X\beta \in \mathfrak{h}_{kj} \wedge C \in \mathrm{core}(X) \wedge [\![C]\!] \Rightarrow [\![A]\!]\}$;
end loop;
$\mathfrak{h}_{jj} := M$;
end loop;

4. Correctness of the recognition algorithm

In this section we will prove a series of lemmas whose combination will constitute an inductive proof of the correctness of our recognition algorithm. First, we do away with the elements of the main diagonal.

Lemma 4.1

(i) $P(\varepsilon, \varepsilon, A \to \rho.\sigma)$ *iff* $A \to \rho.\sigma \in \Pi \wedge \varepsilon \in \rho \wedge [\![S]\!] \Rightarrow [\![A]\!]$,

(ii) $H(\varepsilon, \varepsilon) = \{A \to \rho.\sigma \mid \varepsilon \in \rho \wedge [\![S]\!] \Rightarrow [\![A]\!]\}$,

(iii) $P(z, \varepsilon, A \to \rho.\sigma) \wedge z \neq \varepsilon$ *iff* $A \to \rho\sigma \in \Pi \wedge \varepsilon \in \rho \wedge \exists u, v, \alpha, B, X, \beta, C:$
$z = uv \wedge v \neq \varepsilon \wedge P(u, v, B \to \alpha.X\beta) \wedge C \in \mathrm{core}(X) \wedge [\![C]\!] \Rightarrow [\![A]\!]$,

(iv) $\forall z \neq \varepsilon$: $H(z, \varepsilon) = \{A \to \rho.\sigma \mid \varepsilon \in \rho \wedge \exists u, v, \alpha, B, X, \beta, C:$
$z = uv \wedge v \neq \varepsilon \wedge B \to \alpha.X\beta \in H(u, v) \wedge C \in \mathrm{core}(X) \wedge [\![C]\!] \Rightarrow [\![A]\!]\}$.

Proof. Propositions (i) and (ii) are immediate; (iv) is a rephrasing of (iii). We turn to the proof of (iii).

(*if*): Let $A, \rho, \sigma, u, v, \ldots$ be given accordingly. The definition of P yields η such that

$$[\![S]\!] \overset{*}{\Rightarrow} \eta[\![B]\!] \wedge u \in \eta.$$

Likewise, $[\![B]\!] \overset{*}{\Rightarrow} \alpha[\![C]\!] \wedge v \in \alpha$ and $[\![C]\!] \overset{*}{\Rightarrow} \omega[\![A]\!] \wedge \varepsilon \in \omega$ for some ω. We conclude

$$[\![S]\!] \overset{*}{\Rightarrow} \eta[\![B]\!] \overset{*}{\Rightarrow} \eta\alpha[\![C]\!] \overset{*}{\Rightarrow} \eta\alpha\omega[\![A]\!] \wedge uv \in \eta\alpha\omega$$

so that $P(uv, \varepsilon, A \to \rho.\sigma)$.

(*only if*): By definition we find η such that

$$[\![S]\!] \overset{*}{\Rightarrow} \eta[\![A]\!] \wedge z \in \eta.$$

We may split this derivation into single steps for appropriate A_i, η_i, z_i so that

$$[\![S]\!] = \eta_0[\![A_0]\!] \Rightarrow \eta_0\eta_1[\![A_1]\!] \Rightarrow \cdots \Rightarrow \eta_0\eta_1 \ldots \eta_n[\![A_n]\!] = \eta[\![A]\!]$$

where

$$z = z_1 \ldots z_n \wedge \forall i: \quad z_i \in \eta_i.$$

Let m be the largest index such that $z_i \neq \varepsilon$. For some X, β we obtain

$$A_{m-1} \to \eta_m X\beta \wedge A_m \in \mathrm{core}(X).$$

We conclude

$$z_0 \ldots z_{m-1} \in \eta_0 \ldots \eta_{m-1} \wedge [\![S]\!] \stackrel{*}{\Rightarrow} \eta_0 \ldots \eta_{m-1}[\![A_{m-1}]\!] \wedge z_m \in \eta_m \wedge [\![A_m]\!] \Rightarrow [\![A]\!]$$

so that

$$P(z_0 \ldots z_{m-1}, z_m, A_{m-1} \to \eta_m . X\beta) \wedge [\![A_m]\!] \Rightarrow [\![A]\!].$$

Defining

$$u = z_0 \ldots z_{m-1}, \qquad v = z_m, \qquad B = A_{m-1}, \qquad C = A_m, \qquad \alpha = \eta_m$$

proves the claim. □

The next two lemmas deal with the simpler half of the reduction and shift operators.

Lemma 4.2

(i) $P(u, v, A \to \alpha.X\beta\sigma) \wedge X \Rightarrow a \wedge \varepsilon \in \beta$ *impl* $P(u, va, A \to \alpha X\beta.\sigma)$,

(ii) $H(u, v) + a \subseteq H(u, va)$.

Proof. Immediate. □

Lemma 4.3

(i) $P(u, v, A \to \alpha.B\beta\sigma) \wedge P(uv, w, C \to \delta.) \wedge \varepsilon \in \beta \wedge B \Rightarrow C$

impl $P(u, vw, A \to \alpha B\beta.\sigma)$,

(ii) $P(u, v, A \to \alpha.\langle B \cap C\rangle\beta\sigma) \wedge P(uv, w, D \to \delta.) \wedge P(uv, w, E \to \gamma.)$

$\wedge B \Rightarrow D \wedge C \Rightarrow E \wedge \varepsilon \in \beta$ *impl* $P(u, vw, A \to \alpha\langle B \cap C\rangle\beta.\sigma)$,

(iii) $P(u, v, A \to \alpha.[B]\beta\sigma) \wedge \varepsilon \in \beta \wedge \neg\exists D, \delta: B \Rightarrow D \wedge P(uv, w, D \to \delta.)$

impl $P(u, vw, A \to \alpha[B]\beta.\sigma)$,

(iv) $H(u, v) * H(uv, w) \subseteq H(u, vw)$.

Proof. (iv) is a rephrasing of (i)-(iii). We consider (iii) and leave the first two cases to the reader. The definition of P yields η such that

$$[\![S]\!] \stackrel{*}{\Rightarrow} \eta[\![A]\!] \wedge u \in \eta \wedge v \in \alpha. \qquad (*)$$

We claim $w \in [B]$, i.e. $w \notin B$. In order to prove this claim, suppose $w \in B$. Then there must be some production rule $D \to \delta$ such that $B \Rightarrow D \wedge w \in \delta$. We obtain

$$[\![S]\!] \stackrel{*}{\Rightarrow} \eta[\![A]\!] \stackrel{*}{\Rightarrow} \eta\alpha[\![B]\!] \wedge uv \in \eta\alpha \wedge w \in \delta,$$

that is,

$$B \Rightarrow D \wedge P(uv, w, D \to \delta.).$$

Contradiction! Using (*) and $w \notin B$ we obtain

$$[\![S]\!] \overset{*}{\Rightarrow} \eta[\![A]\!] \wedge u \in \eta \wedge vw \in \alpha[B]\beta,$$

that is,

$$P(u, vw, A \to \alpha[B]\beta.\sigma)$$

as required. □

Unfortunately, the converse of the last two lemmas is much more complicated. We recall that the computation of an off-off-diagonal element $H(u, z)$ proceeds in two phases. In the first phase, sets of the form $H(u, v) * H(uv, w)$ are collected. In the second phase we iterate $H(u, \varepsilon) *_r M$ through the hierarchy of ranks, where M is an approximation to $H(u, z)$. The problem is to characterize the result of the first phase and the addendum of the second phase. The key to the solution for the first phase problem is that the reduction operator $*$ combines the sets corresponding to $vw = z$ where neither v nor w are empty. Taking a simplified view of the reduce operator we see that it takes an item $A \to \alpha.X\beta\sigma$ from $H(u, v)$, derives $w \in X\beta$ from the contents of $H(uv, w)$ and inserts $A \to \alpha X\beta.\sigma$ into $H(u, vw) = H(u, z)$. Consequently, all the items $A \to \rho.\sigma$ added to $H(u, z)$ during the first phase allow a split of ρ into $\rho = \alpha X\beta$ and a corresponding split of z into $z = vw$ such that $v \in \alpha$ and $w \in X\beta$ and neither v nor w are empty. The next lemma will prove that the first phase computes exactly those items which allow a non-trivial split of this kind.

Definition 4.4. The set of *non-trivial splits* of z and $A \to \rho.\sigma$ is defined by

$$\mathrm{NTS}(z, A \to \rho.\sigma) = \{(v, w, \alpha, X, \beta) \mid z = vw \wedge \rho = \alpha X\beta \wedge v \neq \varepsilon \wedge w \neq \varepsilon \wedge v \in \alpha \wedge w \in X \wedge \varepsilon \in \beta\}.$$

Lemma 4.5

(i) $P(u, z, A \to \rho.\sigma) \wedge (v, w, \alpha, a, \beta) \in \mathrm{NTS}(z, A \to \rho.\sigma)$
impl $P(u, v, A \to \alpha.a\beta\sigma) \wedge va = z \wedge v \neq \varepsilon \wedge \varepsilon \in \beta,$

(ii) $P(u, z, A \to \rho.\sigma) \wedge (v, w, \alpha, B, \beta) \in \mathrm{NTS}(z, A \to \rho.\sigma)$
impl $P(u, v, A \to \alpha.B\beta\sigma) \wedge vw = z \wedge v \neq \varepsilon \wedge w \neq \varepsilon \wedge \varepsilon \in \beta$
$\wedge \exists\delta\colon \quad P(uv, w, B \to \delta.),$

(iii) $P(u, z, A \to \rho.\sigma) \wedge (v, w, \alpha, \langle B \cap C\rangle, \beta) \in \mathrm{NTS}(z, A \to \rho.\sigma)$
impl $P(u, v, A \to \alpha.\langle B \cap C\rangle\beta\sigma) \wedge vw = z \wedge v \neq \varepsilon \wedge w \neq \varepsilon \wedge \varepsilon \in \beta$
$\wedge \exists\gamma, \delta\colon \quad P(uv, w, B \to \delta.) \wedge P(uv, w, C \to \gamma.),$

(iv) $P(u, z, A \to \rho.\sigma) \wedge (v, w, \alpha, [B], \beta) \in \mathrm{NTS}(z, A \to \rho.\sigma)$
impl $P(u, v, A \to \alpha[B]\beta\sigma) \wedge vw = z \wedge v \neq \varepsilon \wedge w \neq \varepsilon \wedge \varepsilon \in \beta$
$\wedge \neg\exists\delta\colon \quad P(uv, w, B \to \delta.).$

Proof. ad (i): Assume $P(u, z, A \to \rho.\sigma)$ and let (v, w, α, a, β) be a non-trivial split. By definition of P we find η such that

$$[\![S]\!] \overset{*}{\Rightarrow} \eta [\![A]\!] \wedge u \in \eta.$$

By definition of NTS we have $v \in \alpha$ so that $P(u, v, A \to \alpha.a\beta\sigma)$ is immediate.

ad (ii) From the definitions we obtain

$$A \to \alpha B \beta\sigma \wedge v \in \alpha \wedge w \in B \quad \text{and} \quad [\![S]\!] \overset{*}{\Rightarrow} \eta [\![A]\!] \wedge u \in \eta$$

for some η. We conclude $P(u, v, A \to \alpha.B\beta\sigma)$ and

$$\exists \delta\colon B \to \delta \wedge w \in \delta, \quad \text{that is,} \quad \exists \delta\colon P(uv, w, B \to \delta.).$$

ad (iii): See (ii).

ad (iv): The definitions of P and NTS imply

$$P(u, v, A \to \alpha.[B]\beta\sigma) \wedge vw = z \wedge v \neq \varepsilon \wedge w \neq \varepsilon \wedge \varepsilon \in \beta$$

as well as $w \in [B]$. Suppose $P(uv, w, B \to \delta.)$ for some δ. Then $w \in \delta$ which implies $w \in B$ and $w \notin [B]$. Contradiction! □

Lemma 4.6

(i) $P(u, z, A \to \rho.\sigma) \wedge (y, w, \alpha, a, \beta) \in \mathrm{NTS}(z, A \to \rho.\sigma)$

impl $ya = z \wedge y \neq \varepsilon \wedge A \to \rho.\sigma \in H(u, y) + a,$

(ii) $P(u, z, A \to \rho.\sigma) \wedge (v, w, \alpha, X, \beta) \in \mathrm{NTS}(z, A \to \rho.\sigma)$

impl $vw = z \wedge v \neq \varepsilon \wedge w \neq \varepsilon \wedge A \to \rho.\sigma \in H(u, v) * H(uv, w).$

Proof. By cases using Lemma 4.5. □

The next definition introduces a name for the items covered by this lemma. It will allow us to describe the result of the first phase in the computation of off-off-diagonal elements.

Definition 4.7. The *non-trivial reduction set* of u and z is defined by

$$\mathrm{NRS}(u, z) = \{A \to \rho.\sigma \mid P(u, z, A \to \rho.\sigma) \wedge \mathrm{NTS}(z, A \to \rho.\sigma) \neq \emptyset\}.$$

Theorem 4.8. *Assume* $z = ya \wedge y \neq \varepsilon$. *Then we have*

$$\mathrm{NRS}(u, z) = H(u, y) + a \cup \bigcup \{H(u, v) * H(uv, w) \mid vw = z \wedge v \neq \varepsilon \wedge w \neq \varepsilon\}.$$

Proof. Lemmas 4.2(ii), 4.3(iv) and 4.6. □

Our next goal will be to describe the items which are added in phase two of the computation of off-off-diagonal elements. It would be too simplistic to assume that the phase two items are the ones which do not allow a non-trivial split. Items of phase one may very well take part in phase two, i.e. items which allow a non-trivial split may very well allow trivial splits, too. Therefore, we will characterize the phase two items in a positive way by introducing the notion of a *trivial split*. In addition, we have to partition the phase two items according to ranks in order to cope with the individual steps of our algorithm.

Definition 4.9. The set of *trivial splits* of $A \to \rho.\sigma$ and z is defined by

$$\mathrm{TS}(z, A \to \rho.\sigma) = \{(\alpha, X, \beta) \mid \alpha X \beta = \rho \wedge \varepsilon \in \alpha\beta \wedge z \in X\}.$$

The *level t reduction set* for u and z is defined by

$$\mathrm{RS}(t, u, z) = \mathrm{NRS}(u, z) \cup \{A \to \rho.\sigma \mid P(u, z, A \to \rho.\sigma) \wedge \exists \lambda, X, \mu\colon \ (\lambda, X, \mu) \in \mathrm{TS}(z, A \to \rho.\sigma) \wedge \mathrm{rank}(X) \leqslant t\}.$$

We deal with some trivial cases first.

Theorem 4.10

(i) $\mathrm{RS}(0, u, a) = \{A \to \alpha a\beta.\sigma \mid A \to \alpha.a\beta\sigma \in H(u, \varepsilon) \wedge \varepsilon \in \beta\}$,

(ii) $\mathrm{RS}(0, u\ z) = \mathrm{NRS}(u, z)$ *if* $|z| \geqslant 2$,

(iii) $\mathrm{RS}(h, u, z) = H(u, z)$ *for all* $z \neq \varepsilon$.

Proof. ad (i): There is no non-trivial split for the terminal symbol a so that $\mathrm{NRS}(u, a)$ is empty (see Definitions 4.4 and 4.7). The remaining claim is that the right hand side of (i) equals

$$\{A \to \rho.\sigma \mid P(u, a, A \to \rho.\sigma) \wedge \exists \lambda, X, \mu\colon \ (\lambda, X, \mu) \in \mathrm{TS}(a, A \to \rho.\sigma) \wedge \mathrm{rank}(X) \leqslant 0\}.$$

Expanding TS by its definition while keeping in mind that $\mathrm{rank}(X) \leqslant 0$ implies $X \in \Sigma$ yields

$$\{A \to \alpha a\beta.\sigma \mid P(u, a, A \to \alpha a\beta.\sigma) \wedge \varepsilon \in \beta\}$$

which equals (note the position of the dot)

$$\{A \to \alpha a\beta.\sigma \mid P(u, \varepsilon, A \to \alpha.a\beta\sigma) \wedge \varepsilon \in \beta\}$$

and

$$\{A \to \alpha a\beta.\sigma \mid A \to \alpha.a\beta\sigma \in H(u, \varepsilon) \wedge \varepsilon \in \beta\}$$

as required.

ad (ii): It is sufficient to show that for z with $|z| \geqslant 2$

$$\{A \to \rho.\sigma \mid P(u, z, A \to \rho.\sigma) \wedge \exists \lambda, X, \mu\colon (\lambda, X, \mu) \in \mathrm{TS}(z, A \to \rho.\sigma) \wedge \mathrm{rank}(X) \leqslant 0\} = \emptyset.$$

Again, $\mathrm{rank}(X) \leq 0$ implies $X \in \Sigma$, so that $z \in X$ implies $|z| = 1$. Hence, the condition in the set cannot be satisfied and the set must be empty.

ad (iii): Immediate for the hierarchy constant h. □

Lemma 4.11

(i) $P(u, \varepsilon, A \to \alpha.X\beta\sigma) \wedge z \in B \wedge B \in \mathrm{core}(X) \wedge \mathrm{rank}(X) = t$

impl $\exists D, \delta\colon \quad B \Rrightarrow D \wedge D \to \delta. \in \mathrm{RS}(t-1, u, z)$,

(ii) $P(u, z, A \to \rho.\sigma) \wedge (\alpha, B, \beta) \in \mathrm{TS}(z, A \to \rho.\sigma) \wedge \mathrm{rank}(B) = t$

impl $P(u, \varepsilon, A \to \alpha.B\beta\sigma) \wedge \exists D, \delta\colon \quad B \Rrightarrow D \wedge D \to \delta. \in \mathrm{RS}(t-1, u, z)$,

(iii) $P(u, z, A \to \rho.\sigma) \wedge (\alpha, \langle B \cap C\rangle, \beta) \in \mathrm{TS}(z, A \to \rho.\sigma) \wedge \mathrm{rank}(\langle B \cap C\rangle) = t$

impl $P(u, \varepsilon, A \to \alpha.\langle B \cap C\rangle\beta\sigma)$

$\wedge \exists D, \delta\colon \quad B \Rrightarrow D \wedge D \to \delta. \in \mathrm{RS}(t-1, u, z)$

$\wedge \exists D, \delta\colon \quad C \Rrightarrow D \wedge D \to \delta. \in \mathrm{RS}(t-1, u, z)$,

(iv) $P(u, z, A \to \rho.\sigma) \wedge (\alpha, [B], \beta) \in \mathrm{TS}(z, A \to \rho.\sigma) \wedge \mathrm{rank}([B]) = t$

impl $P(u, \varepsilon, A \to \alpha.[B]\beta\sigma) \wedge \neg\exists D, \delta\colon \quad B \Rrightarrow D \wedge D \to \delta. \in \mathrm{RS}(t-1, u, z)$.

Proof. Recall $\mathrm{rank}(B) = \mathrm{rank}([B])$ and $\mathrm{rank}(\langle B \cap C\rangle) = \max\{\mathrm{rank}(B), \mathrm{rank}(C)\}$.

ad (i): In order to find D and δ as required by our claim we look at derivations starting with B and consider two cases.

Case 1: There is a derivation

$$B = \eta_1 B_1 \omega_1 \Rightarrow \eta_1\eta_2 B_2\omega_2\omega_1 \Rightarrow \cdots \Rightarrow \eta_1 \ldots \eta_n B_n \omega_n \ldots \omega_1 \Rightarrow \eta_1 \ldots \eta_n \lambda X \mu \omega_n \ldots \omega_1$$

where

$$\varepsilon \in \eta_1 \ldots \eta_n\lambda \wedge z \in X \wedge \varepsilon \in \mu\omega_n \ldots \omega_1 \wedge X \in \Phi \setminus N.$$

Then

$$B \Rrightarrow B_n \wedge (\lambda, X, \mu) \in \mathrm{TS}(z, B_n \to \lambda X\mu.) \wedge \mathrm{rank}(X) < \mathrm{rank}(B)$$

and

$$[\![B]\!] \overset{*}{\Rightarrow} \eta_1 \ldots \eta_n [\![B_n]\!] \quad \wedge \quad \varepsilon \in \eta_1 \ldots \eta_n \quad \wedge \quad z \in \lambda X\mu,$$

i.e.

$$P(u, z, B_n \to \lambda X\mu.) \quad \text{and} \quad B_n \to \lambda X\mu. \in \mathrm{RS}(\mathrm{rank}(B) - 1, u, z).$$

Case 2: There is no such derivation. Then there is a derivation

$$B = \eta_1 B_1 \omega_1 \Rightarrow \eta_1\eta_2 B_2\omega_2\omega_1 \Rightarrow \cdots \Rightarrow \eta_1 \ldots \eta_n B_n \omega_n \ldots \omega_1 \Rightarrow \eta_1 \ldots \eta_n \lambda X \mu \omega_n \ldots \omega_1$$

where

$$\varepsilon \in \eta_1 \ldots \eta_n \wedge v \in \lambda \wedge w \in X \wedge v \neq \varepsilon \wedge w \neq \varepsilon \wedge vw = z \wedge \varepsilon \in \mu\omega_n \ldots \omega_1.$$

Then

$$B \Rightarrow B_n \wedge (v, w, \lambda, X, \mu) \in \mathrm{NTS}(z, B_n \to \lambda X \mu.)$$

and

$$[\![B]\!] \overset{*}{\Rightarrow} \eta_1 \ldots \eta_n [\![B_n]\!] \quad \wedge \quad \varepsilon \in \eta_1 \ldots \eta_n \wedge z \in \lambda X \mu,$$

i.e.

$$P(u, z, B_n \to \lambda X \mu.) \quad \text{and} \quad B_n \to \lambda X \mu. \in \mathrm{NRS}(u, z).$$

ad (ii): It is easy to see that the assumptions imply

$$P(u, \varepsilon, A \to \alpha.B\beta\sigma) \wedge z \in B.$$

The rest follows using (i).

ad (iii): See (ii).

ad (iv): As above, the assumptions imply

$$P(u, \varepsilon, A \to \alpha.[B]\beta\sigma) \wedge z \in [B].$$

The other half of the conjunction is proven by contradiction. Suppose, there are D, δ such that

$$B \Rightarrow D \wedge D \to \delta. \in \mathrm{RS}(t-1, u, z).$$

Then, $z \in D$ and, hence $z \in B$ and $z \notin [B]$ are immediate. Contradiction! □

Lemma 4.12

(i) $P(u, \varepsilon, A \to \alpha.B\beta\sigma) \wedge \varepsilon \in \beta \wedge \mathrm{rank}(B) = t$

$\wedge \exists D, \delta: \quad B \Rightarrow D \wedge D \to \delta. \in \mathrm{RS}(t-1, u, z)$

impl $A \to \alpha B \beta.\sigma \in \mathrm{RS}(t, u, z)$,

(ii) $P(u, \varepsilon, A \to \alpha.\langle B \cap C \rangle \beta\sigma) \wedge \varepsilon \in \beta \wedge \mathrm{rank}(\langle B \cap C \rangle) = t$

$\wedge (\exists D, \delta: \quad B \Rightarrow D \wedge D \to \delta. \in \mathrm{RS}(t-1, u, z))$

$\wedge (\exists D, \delta: \quad C \Rightarrow D \wedge D \to \delta. \in \mathrm{RS}(t-1, u, z))$

impl $A \to \alpha \langle B \cap C \rangle \beta.\sigma \in \mathrm{RS}(t, u, z)$,

(iii) $P(u, \varepsilon, A \to \alpha.[B]\beta\sigma) \wedge \varepsilon \in \beta \wedge \mathrm{rank}([B]) = t$

$\wedge \neg \exists D, \delta: \quad B \Rightarrow D \wedge D \to \delta. \in \mathrm{RS}(t-1, u, z)$

impl $A \to \alpha [B] \beta.\sigma \in \mathrm{RS}(t, u, z)$.

Proof. While (i) and (ii) are immediate by Lemma 4.3(i, ii), proposition (iii) is slightly more complicated. In view of Lemma 4.3(iii) it is sufficient to prove that

$$\neg \exists D, \delta: B \Rightarrow D \quad \wedge \quad D \to \delta. \in \mathrm{RS}(t-1, u, z)$$

implies

$$\neg\exists D, \delta\colon B \Rrightarrow D \ \wedge\ P(u, z, D \to \delta.).$$

Contraposition reduces the problem to proving that

$$\exists D, \delta\colon B \Rrightarrow D \ \wedge\ D \to \delta. \in \mathrm{RS}(t-1, u, z) \qquad (*)$$

is implied by

$$B \Rrightarrow D \ \wedge\ P(u, z, D \to \delta.).$$

Suppose that this last formula is fulfilled. It implies $z \in \delta$ and $z \in B$. Adding

$$P(u, \varepsilon, A \to \alpha.[B]\beta\sigma) \ \wedge\ \varepsilon \in \beta$$

we may apply Lemma 4.11(i) to prove (*). □

Theorem 4.13. *For all $t > 0$ and $z \neq \varepsilon$ we have*

$$\mathrm{RS}(t, u, z) = \mathrm{RS}(t-1, u, z) \cup H(u, \varepsilon) *_t \mathrm{RS}(t-1, u, z).$$

Proof. We prove "$\subseteq$" first. Suppose $A \to \rho.\sigma \in \mathrm{RS}(t, u, z) \backslash \mathrm{RS}(t-1, u, z)$. Then we find (α, X, β) such that

$$P(u, z, A \to \rho.\sigma) \wedge (\alpha, X, \beta) \in \mathrm{TS}(z, A \to \rho.\sigma) \wedge \mathrm{rank}(X) = t$$

and

$$A \to \rho.\sigma = A \to \alpha X \beta.\sigma \wedge \varepsilon \in \beta.$$

Lemma 4.11(ii)-(iv) yields

$$P(u, \varepsilon, A \to \alpha.X\beta\sigma), \text{ that is, } A \to \alpha.X\beta \in H(u, \varepsilon)$$

and

$$\begin{aligned} X = B \ \mathit{impl}\ & \exists D, \delta\colon B \Rrightarrow D \ \wedge\ D \to \delta. \in \mathrm{RS}(t-1, u, z) \\ \wedge\, X = \langle B \cap C \rangle \ \mathit{impl}\ & \exists D, \delta\colon B \Rrightarrow D \ \wedge\ D \to \delta. \in \mathrm{RS}(t-1, u, z) \\ & \wedge \exists D, \delta\colon C \Rrightarrow D \ \wedge\ D \to \delta. \in \mathrm{RS}(t-1, u, z) \\ \wedge\, X = [B] \ \mathit{impl}\ & \neg\exists D, \delta\colon B \Rrightarrow D \ \wedge D \to \delta. \in \mathrm{RS}(t-1, u, z) \end{aligned}$$

so that Definition 3.4 of $*_t$ gives us

$$A \to \alpha X \beta.\sigma \in H(u, \varepsilon) *_t \mathrm{RS}(t-1, u, z)$$

as required.

We turn to the proof of "$\supseteq$" and start with

$$A \to \alpha X \beta.\sigma \in H(u, \varepsilon) *_t \mathrm{RS}(t-1, u, z).$$

Using Definition 3.4 of $*_t$ we may safely assume

$$A \to \alpha.X\beta\sigma \in H(u, \varepsilon), \text{ that is, } P(u, \varepsilon, A \to \alpha.X\beta\sigma)$$

and $\varepsilon \in \beta \wedge \text{rank}(X) = t$ as well as

$$X = B \text{ impl } \exists D, \delta\colon B \Rightarrow\!\!\!\Rightarrow D \wedge D \to \delta. \in \text{RS}(t-1, u, z)$$

$$\wedge X = \langle B \cap C \rangle \text{ impl } \exists D, \delta\colon B \Rightarrow\!\!\!\Rightarrow D \wedge D \to \delta. \in \text{RS}(t-1, u, z)$$

$$\wedge \exists D, \delta\colon C \Rightarrow\!\!\!\Rightarrow D \wedge D \to \delta. \in \text{RS}(t-1, u, z)$$

$$\wedge X = [B] \text{ impl } \neg\exists D, \delta\colon B \Rightarrow\!\!\!\Rightarrow D \wedge D \to \delta. \in \text{RS}(t-1, u, z)$$

so that Lemma 4.12 gives us $A \to \alpha X\beta.\sigma \in \text{RS}(t, u, z)$ as required. □

Now all the facts required to prove Algorithm 3.6 have been assembled. In the following theorem all the pieces are put in their proper places.

Theorem 4.14. *Assume* $u = a_1 \ldots a_i$ *and* $v = a_{i+1} \ldots a_j$. *Then*

(i) $\mathfrak{h}_{ij} = H(u, v)$ *for all* $\mathfrak{h}_{ij}$ *computed by Algorithm* 3.6;

(ii) *for all* $w = a_1 \ldots a_n$ *we have*

$$w \in S \text{ iff } \exists\sigma\colon S \to \sigma. \in \mathfrak{h}_{0n};$$

(iii) *Algorithm* 3.6 *operates in cubic time.*

Proof. (ii) is an immediate consequence of (i). Note that the size of item sets and, therefore, the time required for a reduction or shift operation is bounded by a constant. Since the maximum nesting of loops is 3, this proves (iii). For the proof of (i) we present a version of Algorithm 3.6 which makes explicit the references to the lemmas and theorems of this section.

— Initialization: compute $\mathfrak{h}_{00} = H(\varepsilon, \varepsilon)$;
$\mathfrak{h}_{00} := \{A \to \alpha.\beta \mid [\![S]\!] \Rightarrow\!\!\!\Rightarrow [\![A]\!] \wedge \varepsilon \in \alpha\}$;
for $j = 1, 2, 3, \ldots, n$ **loop**
  $u := a_1 \ldots a_{j-1}$; $a := a_j$;
  — off-diagonal: Note $\mathfrak{h}_{j-1,j-1} = H(u, \varepsilon)$ and compute $\mathfrak{h}_{j-1,j} = H(u, a)$
  $M := \{A \to \alpha a\beta.\sigma \mid A \to \alpha.a\beta\sigma \in H(u, \varepsilon) \wedge \varepsilon \in \beta\}$;
  assert $M = \text{RS}(0, u, a)$ by Theorem 4.10;
  for $t = 1, 2, \ldots, h$ **loop**
    $M := M \cup H(u, \varepsilon) *_t M$;
    assert $M = \text{RS}(t, u, z)$ by Theorem 4.13;
  end loop;
  assert $M = H(u, a)$ by Theorem 4.10;
  $\mathfrak{h}_{j-1,j} := M$;
  for $i = j-2, j-3, \ldots, 0$ **loop**

$u := a_1 \dots a_i$; $z := a_{i+1} \dots a_j$; $y := a_{i+1} \dots a_{j-1}$; $a := a_j$;
— off-off-diagonal: Note $\mathfrak{h}_{ij-1} = H(u, y)$ and compute $\mathfrak{h}_{ij} = H(u, z)$;
$M := H(u, y) + a$;
for $k = i+1, i+2, \dots, j-1$ **loop**
  $v := a_{i+1} \dots a_k$; $w := a_{k+1} \dots a_j$;
  — Note $\mathfrak{h}_{ik} = H(u, v)$, $\mathfrak{h}_{kj} = H(uv, w)$
  $M := M \cup H(u, v) * H(uv, w)$;
end loop;
assert $M = \mathrm{NRS}(u, z)$ by Theorem 4.8;
assert $M = \mathrm{RS}(0, u, z)$ by Theorem 4.10;
for $t = 1, 2, \dots, h$ **loop**
  $M := M \cup H(u, \varepsilon) *_t M$; — Note $\mathfrak{h}_{ij} = H(u, \varepsilon)$
  assert $M = \mathrm{RS}(t, u, z)$ by Theorem 4.13;
end loop;
assert $M = H(u, z)$ by Theorem 4.10;
$\mathfrak{h}_{ij} := M$;

end loop;
$z := a_1 \dots a_i$;
— On-diagonal: compute $\mathfrak{h}_{ii} = H(z, \varepsilon)$;
$M := \emptyset$;
for $k = 0, 1, \dots, j-1$ **loop**
  $u := a_1 \dots a_k$; $v := a_{k+1} \dots a_j$; — Note $\mathfrak{h}_{kj} = H(u, v)$
  $M := M \cup \{A \to \rho.\sigma \mid \varepsilon \in \rho \wedge \exists B, \alpha, X, \beta, C:$
    $B \to \alpha.X\beta \in H(u, v) \wedge C \in \mathrm{core}(X) \wedge [\![C]\!] \Rightarrow [\![A]\!]\}$;
end loop;
assert $M = H(z, \varepsilon)$ by Theorem 4.1(iv);
$\mathfrak{h}_{ii} := M$;
end loop; □

5. Concluding remarks

Our shift and reduction operators go back to similar operators introduced by Harrison in [3]. Their purpose is both to simplify the presentation of the algorithm and its proof and to bypass ε and chain productions, thus increasing efficiency. Indeed, without such operators the proof of our extended algorithm might have become unmanageably complex. On the other hand, removing from Section 4 all the parts pertaining to complementation and intersection would yield a rather short and formal proof of Earley's algorithm.

Within either of the two k-loops of Algorithm 3.4, the order of computation is quite irrelevant: the iterations might even be done in parallel. A variant of our algorithm developed in a different notational framework [7] allows us to interchange or execute in parallel other parts of the computation.

References

[1] J. Earley, An efficient context-free parsing algorithm, *Comm. ACM* **13** (1970) 94–102.
[2] S.L. Graham, M.A. Harrison and W.L. Ruzzo, An improved context-free recognizer, *ACM TOPLAS* **2** (1980) 415–462.
[3] M.A. Harrison, *Introduction to Formal Language Theory* (Addison-Wesley, Reading, MA, 1978).
[4] R. McNaughton and H. Yamada, Regular expressions and state graphs for automata, *IRE Trans. Electron. Comput.* (1960) 39–47.
[5] A.K. Salomaa, *Formal Languages* (Academic Press, New York, 1973).
[6] L. Valiant, General context-free recognition in less than cubic time, *J. JCCS* **10** (1975) 308–315.
[7] L. Schmitz, Adapting the Earley–Harrison algorithm to a class of non-context-free grammars, Techn. Rep. 8608 Fakultät für Informatik, Universität der Bundeswehr, München (1986).